Homework 1: Perception
ME 5413 Autonomous Mobile Robot

Anse Min

A0285307B

(Master of Science (Robotics), NUS)

A REPORT SUBMITTED FOR THE ASSIGNMENT OF ME5413

DEPARTMENT OF MECHANICAL ENGINEERING

NATIONAL UNIVERSITY OF SINGAPORE

Supervisor:

Prof. Marcelo H Ang Jr

22/2/2024

1. Introduction

This project delves into advanced object tracking and trajectory prediction within computer vision, utilizing two distinct datasets: one comprising 100 JPG images labeled with object positions for tracking evaluation, and another in NPZ format containing spatial-temporal data for scenario analysis. Leveraging technologies such as OpenCV-Python and Python 3.9, alongside visualization tools like Matplotlib and Pandas, the study embarks on a comprehensive examination of tracking techniques, from template matching and Kalman filtering to the integration of Histogram Filtering for enhanced accuracy. Task 2 extends the exploration to trajectory prediction, employing models of constant velocity and acceleration to predict future positions, assessed via Average Displacement Error (ADE) and Final Displacement Error (FDE) metrics. This project aims to refine object tracking and prediction methodologies, offering insights into their application in real-world scenarios.

2. Data Specifications and System Requirements

The project utilizes a dataset consisting of 100 JPG images from varied activities such as running, dancing, and sports, labeled with initial and actual object positions for tracking evaluation (Figure 1). A second dataset in NPZ format contains spatial-temporal data like lanes and crosswalks for advanced trajectory and scenario analysis, with a PNG image for visualization. The system requirements include OpenCV-Python (headless version) for image processing, Python 3.9 for compatibility with modern libraries, and visualization tools like Matplotlib and Pandas, highlighting the project's backend processing and data analysis focus.

Figure 1: Dataset (Sequence 1 (left) to 5 (right) respectively.

3. Task 1

Task 1 encompasses a comprehensive exploration of object-tracking techniques through computer vision and algorithmic methods. Initially, it delves into template matching using various OpenCV techniques to track objects across image sequences, highlighting the process from initialization to visualization. Subsequently, it employs a Kalman filter for more refined object tracking, comparing predicted positions with actual ground truth data across multiple sequences. The project rigorously evaluates the performance of both methods by calculating Intersection over Union (IoU) and precision metrics, providing a detailed comparison of their effectiveness. To address the limitations identified through these evaluations, a proposal suggests integrating Histogram Filtering with the Kalman filter. This approach aims to enhance tracking accuracy by adapting to variations in object appearance and providing more accurate measurements for the Kalman filter, thereby improving the system's applicability to real-world scenarios.

3.1. Template Matching

The implementation of template matching in this project leverages a variety of OpenCV methods to track objects through a series of images. The core concept of template matching is to take a smaller image or template—extracted from the initial frame using the coordinates provided in 'first track.txt'—and slide it across the search area in subsequent frames to find the best possible match based on predefined criteria. Different matching methods are employed to cater to a range of scenarios and to account for variations in lighting, scale, and orientation. These methods include `TM_CCOEFF`, which correlates the template over the image; `TM_CCOEFF_NORMED`, which normalizes the results, allowing for better comparison across different scales and rotations; `TM_CCORR_NORMED`, focusing on the correlation ratio; `TM_SQDIFF`, which uses the square difference between the template and image patches; and `TM_SQDIFF_NORMED`, which is a normalized version of the square difference method.

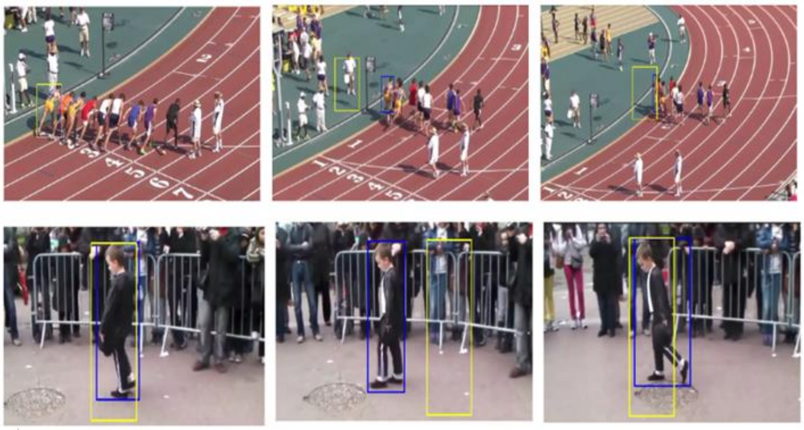

Figure 2: Sequence 1 (up) and 2(down)

In the code, an iterative process runs through each image in the sequence, applying each method to find the location where the template matches best. The script calculates the match's coordinates—the top-left corner of the bounding box—and records these in a results file for every frame. The iterative nature of this process, applied across the entire sequence, ensures a comprehensive assessment of each method's efficacy in real-time object tracking. The result of this process is a series of coordinates that map the trajectory of the object over time. The effectiveness of the matching is later verified by visualizing the matched regions against the ground truth bounding boxes, as indicated in Figure 2 and Figure 3. This visualization is a key step as it provides an immediate visual assessment of each method's accuracy in tracking the object throughout the sequence. By comparing the predicted and actual locations of the object, it's possible to fine-tune the matching algorithms or choose the most effective one for the given context.

Figure 3: Sequence 3 (up) to 5 (down)

3.2. Kalman Filter

The Kalman filter implementation in this task is an exercise in applying predictive analytics to the domain of object tracking in image sequences. The Kalman filter algorithm is adept at handling uncertainty and noise in dynamic systems, making it well-suited for the task of tracking an object whose position changes over time. The code begins with the crucial step of initializing the Kalman filter using the initial position and size of the object, which sets the stage for the subsequent prediction and estimation processes. The filter is configured with a state transition matrix that encapsulates the assumption of constant velocity motion—a common model in object tracking—which helps in predicting the object's future positions. The process noise covariance and measurement noise covariance matrices are fine-tuned to balance the process dynamics against the measurement reliability, enabling the filter to adjust its estimates in response to the actual observed movements of the object. As each new frame arrives, the algorithm engages in a two-step process: prediction and correction. In the prediction step, the Kalman filter uses the model of the object's motion to estimate its next position. This prediction is then refined in the correction step, where the actual observed position (taken from the ground truth data) is used to update the model. This iterative process allows the filter to maintain a dynamic model of the object's motion that is responsive to changes in trajectory or speed. The visualization of this is shown below in Figures 4 and 5.

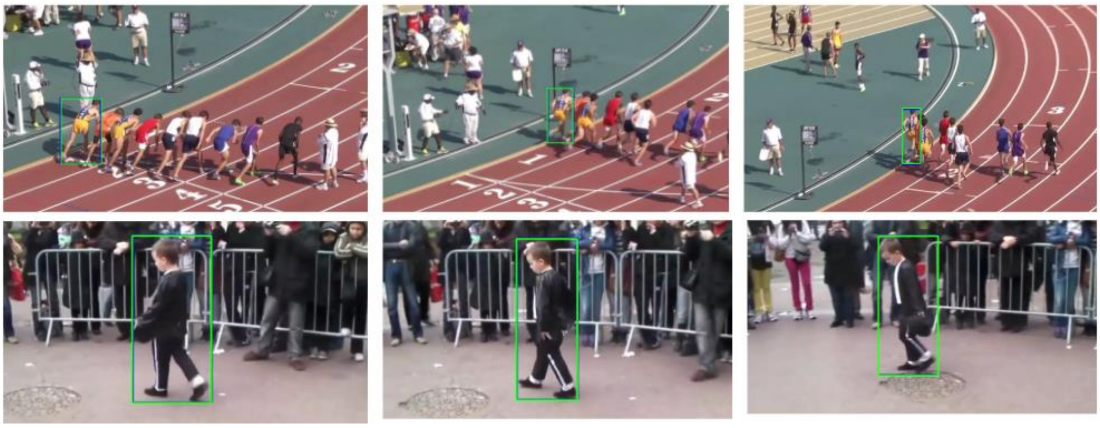

Figure 4: Sequence 1 (up) and 2 (down)

Figure 5: Sequence 3 (up) to 5 (down)

3.3. Performance Evaluation and Visualization

The performance of object tracking algorithms is comprehensively assessed through two distinct implementations: template matching and Kalman Filter, using Intersection over Union (IoU) and precision metrics. Template matching methods (`TM_CCOEFF`, `TM_CCOEFF_NORMED`, `TM_CCORR_NORMED`, `TM_SQDIFF`, `TM_SQDIFF_NORMED`) are evaluated based on the accuracy of overlap between predicted and ground truth bounding boxes, with IoU aiming for scores close to 1. Precision measures how closely the center of the predicted box aligns with the ground truth, with lower distances indicating higher precision. These metrics are calculated for each method across sequences and averaged to gauge object tracking accuracy.

In contrast, the Kalman Filter approach predicts future object states based on a constant velocity model, continuously refining these predictions by correcting against ground truth data. This dynamic model of tracking, bolstered by its predictive and corrective capabilities, is also evaluated using IoU and precision metrics to provide a dual-metric assessment of tracking accuracy. The results are then aggregated to determine the average performance across sequences. The Kalman Filter consistently demonstrates superior tracking effectiveness, accurately following objects across various scenarios, as evidenced by higher average IoU scores and lower precision distances. This integrated evaluation reveals the strengths of the Kalman Filter over template matching methods, particularly its robustness in handling the temporal aspects of tracking and adapting to the dynamic changes in object movements.

We can then compare the average precision and IoU of the Kalman filter and template-matching implementation in Figure 6. The Kalman Filter's superior tracking accuracy over template matching methods is underscored by its predictive capabilities, dynamic correction, robustness to noise, and consistency across sequences, as indicated by higher IoU and lower precision scores in the bar charts. Unlike template matching, which processes each frame in isolation, the Kalman Filter forecasts an object's future position using a state model and refines these predictions with each new observation. This state model grants it inherent accuracy that static template matching methods cannot match because its prediction was based on the ground truth data. The Kalman Filter's performance is less susceptible to variations in object appearance and more adaptive to changes within the tracked sequence. Its reliance on ground truth for corrections makes the Kalman Filter an inherently more accurate system, offering a dynamic and adaptive solution for object tracking that outperforms the relatively rigid template matching techniques.

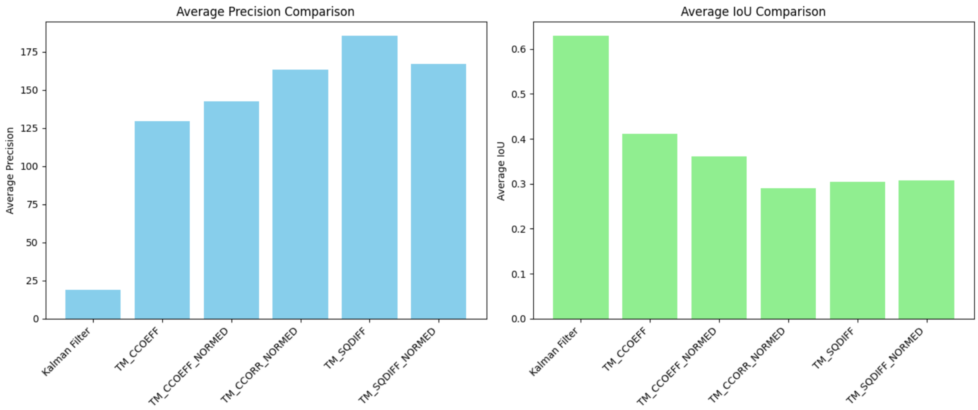

Figure 6: Kalman filter vs Template matching.

Lastly, the potential improvement for object tracking involves integrating Histogram Filtering with the Kalman Filter to address current limitations of accuracy in varying conditions and the unrealistic reliance on ground truth data. Histogram Filtering uses color distribution to locate objects, making it more resilient to appearance changes than template matching. By feeding Histogram Filtering measurements into the Kalman Filter, the system gains adaptability and enhances precision by smoothing measurement noise. This proposed integration not only increases tracking robustness but also better suits real-world applications where ground truth is unavailable, offering a more dynamic and realistic tracking solution.

4. Task 2

4.1. Constant velocity ADE and FDE calculations

This implementation facilitates trajectory prediction and evaluation in a multi-object tracking environment by predicting future agent positions using a constant velocity model and assessing the predictions with Average Displacement Error (ADE) and Final Displacement Error (FDE) against ground truth. The process begins with loading and preprocessing data from a `.npz` file, selecting agent indices for prediction, and establishing prediction horizons. The `predict_trajectory_constant_velocity` function then predicts future positions based on the latest known velocity and position, assuming this velocity remains constant over time. The accuracy of these predictions is quantitatively measured by ADE—the mean Euclidean distance between predicted and actual positions over all timesteps—and FDE—the Euclidean distance at the final timestep, both averaged across agents. These metrics provide insight into the performance of the constant velocity prediction model, with the script outputting ADE and FDE values for comprehensive evaluation.

4.2. Constant acceleration ADE and FDE calculation

This part of the implementation employs a Constant Acceleration Model (CAM) to predict the future positions of multiple agents and evaluates the accuracy of these predictions using Average Displacement Error (ADE) and Final Displacement Error (FDE). Starting with estimating acceleration based on velocity changes in the last two timesteps, the code calculates the agents' accelerations in the x and y directions. Leveraging this data, the `predict_trajectory_constant_acceleration` function forecasts future positions for various prediction horizons by applying the principles of constant acceleration motion. The predictions are then rigorously assessed against the ground truth. ADE is calculated as the mean Euclidean distance between predicted and actual positions across all timesteps and agents, providing an average accuracy measure. FDE measures the Euclidean distance at the final predicted timestep, offering an endpoint accuracy assessment. These metrics are computed for each prediction horizon, with the results indicating the precision of the constant acceleration prediction model in tracking agent movement over time.

4.3. Discussion

The similar performance of the Constant Acceleration Model (CAM) and Constant Velocity Model (CVM) in predicting future positions might be due to the dataset's agents moving in relatively straight lines with uniform speeds, making acceleration negligible. This is especially true over short prediction horizons where acceleration's impact on position is minimal. Additionally, any small accelerations present may be obscured by noise in the data or by the brief time intervals between samples, which could prevent the models from displaying distinct predictive behaviors. The dataset's lack of complex movement patterns could also contribute to the convergence of ADE and FDE values across both models, indicating the need for more varied data to discern the models' differences effectively.

5. Conclusion

In conclusion, this project demonstrated the application and evaluation of advanced object tracking and trajectory prediction techniques within the field of computer vision. Through the utilization of datasets featuring a variety of activities, the study has explored the effectiveness of template matching, Kalman filtering and template matching. Task 2 further extended the investigation into trajectory prediction, employing constant velocity and acceleration models to predict future positions, evaluated through ADE and FDE metrics. The findings underscore the superiority of the Kalman filter in tracking accuracy and highlight the potential of integrating Histogram Filtering for improved performance in dynamic conditions. This report contributes valuable insights into the development of more robust object-tracking systems for future applications in real-world autonomous mobile robot scenarios.